

Curso Superior de Engenharia da Computação

Relatório de Laboratório 2

Roteiro: Processos e Threads

Turma: 2018.1

Aluno: Erick Spinelli Pimentel

Data: 15/08/2019

**processdemo.c**

2 - A saída é o que você esperava? Justifique.

Sim, pois existe uma contagem que começa com 50(PAI) e outra contagem também começa com 50(Filho), ao executar o programa, é acrescentado 1 a contagem do pai e decrescido 1 na contagem do filho várias vezes(while true).

4 - Qual o processo pai e qual o processo filho? (Dica, verifique a coluna PID e PPID. Se não souber o que é PID e PPID, procure no Google). Justifique.

PID PPID

0 1000 3152 2245 ...

1 1000 3153 3152 …

o processo pai é o 3152(PID) e o processo filho é o 3153(PID). Pois o numero do PPID do processo que possui o PID 3153, é o mesmo número de PID do outro processo.

5 - O processo pai continuou rodando

6 - O processo filho continuou rodando

7 - Não faz diferença matar o filho ou o pai, pois o espaço de endereçamento do pai e o espaço do endereçamento do filho são diferentes (são independentes)

**threaddemo.c**

2 - Foi feito para ser “igual” ao processdemo.c só que com Threads, porém eles estão usando a mesma variável, pois as threads compartilham o espaço de endereçamento do processo (tem um processo, o processo tem um espaço de endereçamento, as threads estão dentro desse espaço de endereçamento, por isso a variável global é compartilhada)

3 - threaddemo.c é mais rápido do que processdemo.c, pois a troca de contexto do processdemo.c é maior do que a troca de contexto de threaddemo.c (o contexto do espaço de endereçamento de uma thread é muito menor do que o contexto do espaço de endereçamento de um processo)

5 - Ele acrescenta 1 apenas uma vez e decresce 1 apenas uma vez, por o loop infinito (while true) foi removido

6 - Solução: criar duas variáveis locais (criar dois espaços de endere;amento local dentro do processo)